CS 6140 Machine Learning
Northeastern University, Fall 2018

Homework 2

Due 11:59 p.m. Friday, Sep 28, 2018

Please submit your written solutions as a PDF file using the provided LaTeX template. (`http://roseyu.com/CS6140/latex_template.tex`).

1 Logistic Regression

Consider the following loss function of logistic regression:

$$L(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N} \log(\sigma(y_i \mathbf{w}^\top \mathbf{x}_i))$$

where $y_i \in \{-1, 1\}$, σ is the sigmoid function.
Q1: what is the Hessian $\mathbf{H}$ of this loss function?
Q2: show $\mathbf{z}^\top \mathbf{H}\mathbf{z} \geq 0$ for any vector $\mathbf{z}$.

2 Bias-Variance Trade-off

Derive the bias-variance decomposition for the squared error loss function. That is, show that for a model f_S trained on a dataset S to predict a target $y(x)$ for each x,

$$\mathbb{E}_S\left[E_{\text{out}}\left(f_S\right)\right] = \mathbb{E}_x[(x) + (x)]$$

given the following definitions:

$$\begin{aligned} F(x) &= \mathbb{E}_S\left[f_S(x)\right] \\ E_{\text{out}}(f_S) &= \mathbb{E}_x\left[\left(f_S(x) - y(x)\right)^2\right] \\ \text{Bias}(x) &= (F(x) - y(x))^2 \\ \text{Var}(x) &= \mathbb{E}_S\left[(f_S(x) - F(x))^2\right] \end{aligned}$$

3 Perceptron

The perceptron is a simple linear model used for binary classification. For an input vector $\mathbf{x} \in \mathbb{R}^d$, weights $\mathbf{w} \in \mathbb{R}^d$, and bias $b \in \mathbb{R}$, a perceptron $f : \mathbb{R}^d \to \{-1, 1\}$ takes the form

$$f(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + b\right)$$

The weights and bias of a perceptron can be thought of as defining a hyperplane that divides $\mathbb{R}^d$ such that each side represents an output class. For example, for a two dimensional dataset, a perceptron could be drawn as a line that separates all points of class $+1$ from all points of class -1.

The PLA (or the Perceptron Learning Algorithm) is a simple method of training a perceptron. First, an initial guess is made for the weight vector $\mathbf{w}$. Then, one misclassified point is chosen arbitrarily and the $\mathbf{w}$ vector is updated by

$$\mathbf{w}_{t+1} = \mathbf{w}_t + y(t)\mathbf{x}(t)$$
$$b_{t+1} = b_t + y(t),$$

where $\mathbf{x}(t)$ and $y(t)$ correspond to the misclassified point selected at the t^{th} iteration. This process continues until all points are classified correctly.
Download the source file (`roseyu.com/CS6140/HW2.zip`) and work with the provided Jupyter notebook, titled `HW2_notebook.ipynb`. This notebook utilizes the file `perceptron_helper.py`, but no modification is needed.

The graph below shows an example 2D dataset. The + points are in the +1 class and the $\circ$ point is in the -1 class.

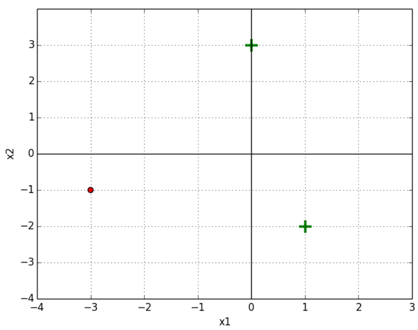


Figure 1: The green + are positive and the red $\circ$ is negative

Q1: Implement the `update_perceptron` and `run_perceptron` methods in the notebook, and perform the perceptron algorithm with initial weights $w_1 = 0, w_2 = 1, b = 0$. Give your solution in the form a table showing the weights and bias at each timestep and the misclassified point $([x_1, x_2], y)$ that is chosen for the next iteration's update. You can iterate through the three points in any order. Your code should output the values in the table below; cross-check your answer with the table to confirm that your perceptron code is operating correctly.

t	b	w_1	w_2	x_1	x_2	y
0	0	0	1	1	-2	+1
1	1	1	-1	0	3	+1
2	2	1	2	1	-2	+1
3	3	2	0			

A dataset $S = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\} \subset \mathbb{R}^d \times \mathbb{R}$ is *linearly separable* if there exists a perceptron that correctly classifies all data points in the set. In other words, there exists a hyperplane that separates positive data points and negative data points.

Q2: In a 2D dataset, how many data points are in the smallest dataset that is not linearly separable, such that no three points are collinear? How about for a 3D dataset such that

no four points are coplanar? Please limit your solution to a few lines - you should justify but not prove your answer.
Finally, how does this generalize for an N-dimensional set, in which **no** $< N$-dimensional hyperplane contains a non-linearly-separable subset? For the N-dimensional case, you may state your answer without proof or justification.

Q3: Run the visualization code in the Jupyter notebook corresponding to question 3. Assume a dataset is *not* linearly separable. Will the Perceptron Learning Algorithm ever converge? Why or why not?

4 Naive Bayes

Recall that in the Naive Bayes algorithm we calculate

$$p(y|\mathbf{x}) \propto p(\mathbf{x}|y)p(y) = p(y)\prod_i p(\mathbf{x}_i|y)$$

In Gaussian Naive Bayes, we learn a one-dimensional Gaussian for each feature in each class, i.e.

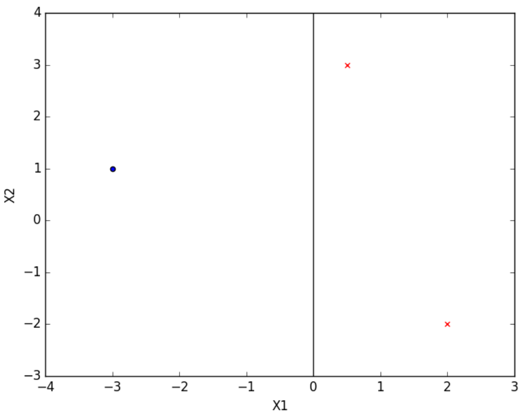
$$p(\mathbf{x}_i|y) = N(\mathbf{x}_i; \mu_{i,y}, \sigma^2_{i,y})$$

where $\mu_{i,y}$ is the mean of feature $\mathbf{x}_i$ for those instances in class y, and $\sigma^2_{i,y}$ is the variance of feature $\mathbf{x}_i$ for instances in class y.

Q1: Derive the maximum likelihood estimator for Gaussian Naive Bayes. Write out the estimator for $\mu_{i,y}$ and $\sigma^2_{i,y}$.

5 Support Vector Machine

In class, we discussed that SVM uses Hinge loss: $L_{\text{hinge}} = \max(0, 1 - y\mathbf{w}^T\mathbf{x})$. Consider the set of points $S = \{(\frac{1}{2}, 3), (2, -2), (-3, 1)\}$ in 2D space, shown below, with labels $(1, 1, -1)$ respectively.



Q1: Given a linear model with weights $w_0 = 0, w_1 = 1, w_2 = 0$ (where w_0 corresponds to the bias term), compute the gradients $\nabla_w L_{\text{hinge}}$ and $\nabla_w L_{\log}$ of the hinge loss and log loss, and calculate their values for each point in S.
The example dataset described above. Positive instances are represented by red x's, while negative instances appear as blue dots.